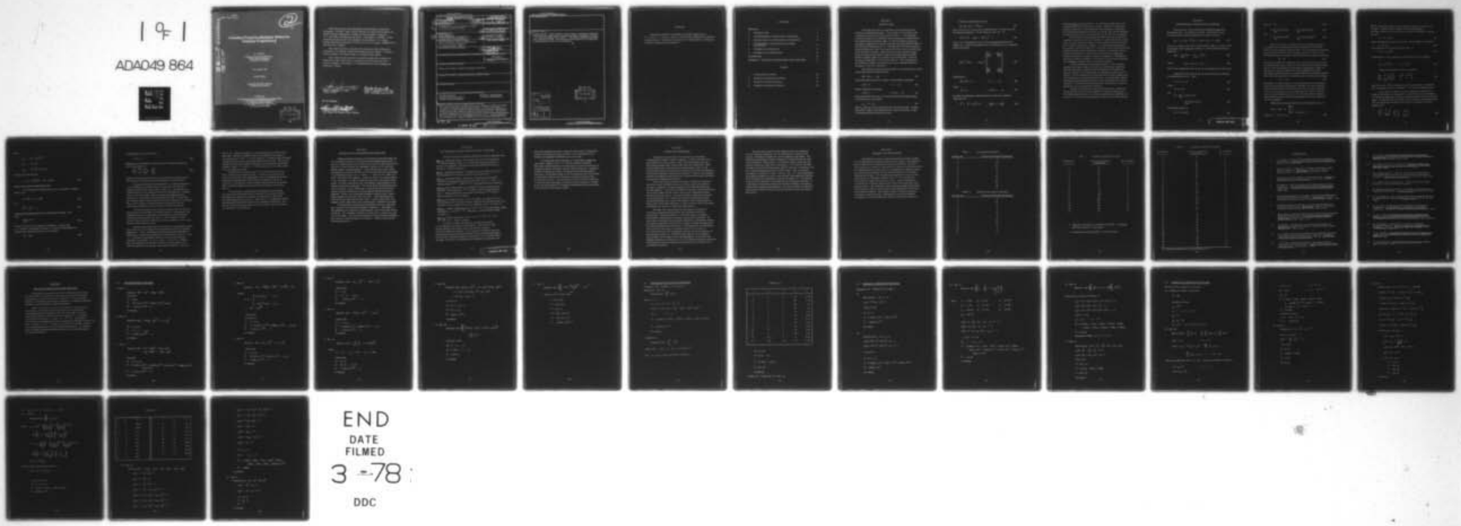
AD-A049 864 AEROSPACE CORP EL SEGUNDO CALIF ENGINEERING SCIENCE --ETC F/G 12/1
A GRADIENT PROJECTION-MULTIPLIER METHOD FOR NONLINEAR PROGRAMMI--ETC(U)
JAN 78 J T BETTS F04701-77-C-0078
UNCLASSIFIED TR-0078(3451-10)-1 SAMSO-TR-78-5 NL

1 OF 1
ADA049 864



END
DATE
FILMED
3-78
DDC

AD A049864

AD No. ___

DDC FILE COPY

REPORT SAMSO-TR-78-5

2

A Gradient Projection-Multiplier Method for Nonlinear Programming

J. T. BETTS

Engineering Science Operations
The Aerospace Corporation
El Segundo, Calif. 90245

30 January 1978

Interim Report

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED

Prepared for
SPACE AND MISSILE SYSTEMS ORGANIZATION
AIR FORCE SYSTEMS COMMAND
Los Angeles Air Force Station
P.O. Box 92960, Worldway Postal Center
Los Angeles, Calif. 90009

This interim report was submitted by The Aerospace Corporation, El Segundo, CA 90245, under Contract F04701-75-C-0076 with the Space and Missile Systems Organization, Deputy for Advanced Space Programs, P.O. Box 92960, Worldway Postal Center, Los Angeles, CA 90009. It was reviewed and approved for The Aerospace Corporation by J. R. Allder, Engineering Science Operations. First Lt Arturo G. Fernandez, SAMSO/YCPT was the project engineer.

This report has been reviewed by the Information Office (OIS) and is releasable to the National Technical Information Service (NTIS). At NTIS, it will be available to the general public, including foreign nations.

This technical report has been reviewed and approved for publication. Publication of this report does not constitute Air Force approval of the report's findings or conclusions. It is published only for the exchange and stimulation of ideas.

Arturo G Fernandez

ARTURO G FERNANDEZ, Lt, USAF
Project Officer

Robert W. Lindemuth

ROBERT W LINDEMUTH, Lt Col, USAF
Chief, Technology Plans Division

FOR THE COMMANDER

Leonard E. Baltzell

LEONARD E BALTZELL, Col, USAF
Asst, Deputy for Advanced Space Programs

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER (18) (19) SAMSO TR-78-5	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER (9) Technical rept.,	
4. TITLE (and Subtitle) (6) A Gradient Projection-Multiplier Method for Nonlinear Programming.	5. TYPE OF REPORT & PERIOD COVERED
	6. PERFORMING ORG. REPORT NUMBER (14) TR-0078(3451-10)-1
7. AUTHOR(s) (10) John T. Betts, Performance Analysis Department, Guidance and Control Division	8. CONTRACT OR GRANT NUMBER(s) (15) F04701-77-C-0078
9. PERFORMING ORGANIZATION NAME AND ADDRESS Space and Missile Systems Organization, Air Force Systems Command, Los Angeles, Calif. 90045	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS	12. REPORT DATE (11) 30 Jan[redacted]78
	13. NUMBER OF PAGES (12) 43 p.
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) Unclassified
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Multiplier Method
Gradient Projection Method
Penalty Function Methods
Nonlinear Programming
Parameter Optimization

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report describes a gradient projection-multiplier method for solving the general nonlinear programming problem. The algorithm poses a sequence of unconstrained optimization problems which are solved using a new projection-like formula to define the search directions. The unconstrained minimization of the augmented objective function determines points where the gradient of the Lagrangian function is zero. Points satisfying the constraints are located by applying an unconstrained algorithm to a

Next page

404 068 Inc

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

19. KEY WORDS (Continued)

20. ABSTRACT (Continued)

penalty function. New estimates of the Lagrange multipliers and basis constraints are made at points satisfying either a Lagrangian condition or a constraint satisfaction condition. The penalty weight is increased only to prevent cycling. The numerical effectiveness of the algorithm is demonstrated on a set of test problems.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

PREFACE

The author gratefully acknowledges the helpful suggestions of W. H. Ailor, J. L. Searcy, and D. A. Schermerhorn during the preparation of this document. The author also thanks D. M. Himmelblau, for supplying a number of interesting test problems.

CONTENTS

TABLES

SECTION 1

INTRODUCTION

This paper describes an algorithm for solving the general nonlinear programming problem. The method is an extension of the accelerated multiplier method described in Ref. 1. The algorithm combines aspects of the multiplier method proposed by Powell (Ref. 2) and independently by Hestenes (Ref. 3), with the gradient projection algorithms suggested by a number of authors including Fletcher (Ref. 4) and Murtagh and Sargent (Ref. 5). The algorithm deals with inequality constraints directly, while retaining the favorable numerical properties of the multiplier methods. Search directions are computed using a projection-like formula which avoids the ill-conditioning in penalty function methods reported by Fletcher and McCann (Ref. 6) and addressed by Biggs (Ref. 7) and Murray (Ref. 8). The convergence of the multiplier method is accelerated by using a gradient projection technique to solve the constraints.

The problem of interest in this paper is to determine the n-vector x that minimizes the scalar function,

$$f(x) = f(x_1, \ldots, x_n) \qquad (1)$$

called the objective function, subject to the equality constraints

$$c_i(x) = 0, \qquad i = 1, \ldots, k, \qquad (2)$$

and the inequality constraints

$$c_i(x) \geq 0 \qquad i = (k+1), \ldots, m. \qquad (3)$$

The functions $f(x)$ and $c_i(x)$ are assumed continuously differentiable to second order in the region

$$x_L \leq x \leq x_U \qquad (4)$$

where x_L and x_U are the specified lower and upper bounds. Bounds determine a region of computability and, unlike constraints, cannot be violated during the iterative process.

Define the Lagrangian function,

$$L(x, \lambda) = f(x) + c^T(x)\,\lambda \tag{5}$$

where c(x) is the m-vector of all constraints and λ is the m-vector of Lagrange multipliers. At the optimum point (x^*, λ^*)

$$\nabla L(x^*, \lambda^*) = g(x^*) + G(x^*)\,\lambda^* = 0 \tag{6}$$

where ∇L is the gradient vector of the Lagrangian function with respect to x, g(x) is the gradient vector of f(x), and the n x m Jacobian matrix is given by

$$G(x) = \left[\,\nabla c_1, \ldots, \nabla c_m\,\right] = \begin{bmatrix} \frac{\partial c_1}{\partial x_1} & \cdots & \frac{\partial c_m}{\partial x_1} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \frac{\partial c_1}{\partial x_n} & & \frac{\partial c_m}{\partial x_n} \end{bmatrix} . \tag{7}$$

Furthermore,

$$\lambda_i^* \, c_i(x^*) = 0 \qquad i = 1, \ldots, m, \tag{8}$$

where

$$\lambda_i^* \le 0 \qquad i = (k+1), \ldots, m. \tag{9}$$

In order to distinguish constraints that are active at a solution, define the set

$$B^* = \left\{ i \mid c_i(x^*) = 0, \qquad i \in \{1, 2, \ldots m\} \right\}, \tag{10}$$

called the basic set of constraints. An estimate B of the basic set of constraints shall be referred to as a basis. Clearly B^* contains all equality constraints. If the gradients of the constraints in the basis are linearly independent at the solution, then (6), (8), and (9) constitute the Kuhn-Tucker necessary conditions for the existence of an optimum.

Most nonlinear programming algorithms proceed by obtaining a sequence of points which either satisfy the criteria (8) which we shall refer to as the constraint condition or the condition (6) which we shall refer to as the Lagrangian condition. Projected gradient algorithms, for example, attempt to satisfy the constraints at each step while moving toward satisfaction of the Lagrangian condition. In contrast, penalty function and multiplier methods attempt to satisfy the Lagrangian condition and then move toward constraint satisfaction. The gradient projection-multiplier method to be described cycles between satisfying the Lagrangian condition and the constraint condition.

Points satisfying the Lagrangian condition or the constraint condition are located using an unconstrained optimization algorithm. When the algorithm is applied to the minimization of an augmented objective function, a point satisfying the Lagrangian condition can be located. A point satisfying the constraints can be found by applying the same algorithm to the penalty function alone. The algorithm requires both function and gradient information and is designed for the class of problems in which the function and gradient evaluations are relatively expensive from a computational standpoint.

Section 2 describes the unconstrained algorithm, and Section 3 discusses how the procedure is applied to constrained optimization. The detailed description of the constrained optimization algorithm is in Section 4. Section 5 presents numerical experience with the approach.

SECTION 2

UNCONSTRAINED OPTIMIZATION ALGORITHM

In this section, the unconstrained optimization algorithm developed in Ref. 1 is modified somewhat. Specifically, we are concerned with the following augmented objective function:

$$J(x, \lambda, r) = f(x) + c^T(x)\,\lambda + rP(x) = L(x, \lambda) + r\,P(x), \tag{11}$$

where r is a scalar referred to as the penalty weight, λ is an m-vector of estimates of the Lagrange multipliers, and the penalty function is defined by

$$P(x) = \sum_{i \in B} c_i^2(x) + \sum_{i \in B'} \tilde{c}_i^{\,2}(x), \tag{12}$$

where

$$\tilde{c}_i(x) = \min\,[0,\ c_i(x)\,], \tag{13}$$

and B', the complement of B, is the set of constraints not in the basis.

Expressions for the first and second derivatives are obtained by differentiation of (11). Thus,

$$\nabla J = \nabla L + r\,\nabla P \tag{14}$$

where

$$\nabla L = g + G\lambda \tag{15}$$

and

$$\nabla P = \sum_{i \in B} 2c_i(x)\,\nabla c_i(x) + \sum_{i \in B'} 2\tilde{c}_i(x)\,\nabla \tilde{c}_i(x). \tag{16}$$

The Hessian matrix is

$$H = T + r\,(U+V), \tag{17}$$

PRECEDING PAGE BLANK

where $T = \nabla^2 L$, (18)

$$U = \sum_{i \in B} 2c_i(x)\nabla^2 c_i(x) + \sum_{i \in B'} 2\tilde{c}_i(x)\nabla^2 \tilde{c}_i(x), \tag{19}$$

and

$$V = \sum_{i \in B} 2\nabla c_i(x)\, \nabla c_i(x)^T + \sum_{i \in B'} 2\nabla \tilde{c}_i(x)\nabla \tilde{c}_i(x)^T. \tag{20}$$

When the function J is approximated by a quadratic function, an estimate of the minimum point can be obtained by locating the minimum of the quadratic approximation. The gradient at the minimum point of the approximation must necessarily be zero, and it can be demonstrated that the gradient condition defines the system of equations

$$Hs = \nabla J. \tag{21}$$

The Hessian matrix H is defined by (17), where the matrices T and U are approximated using a rank-one recursive formula originally stated in Ref. 9 and specialized for least squares applications in Ref. 10. The matrix V can be evaluated from local gradient information. The search direction vector s is usually obtained by solving the system (21), and a new estimate of the optimum point constructed according to

$$\overline{x} = x - \rho s. \tag{22}$$

The scalar ρ is determined by a one-dimensional search procedure. Thus the unconstrained optimization algorithm proceeds by taking a series of steps defined by (22), where at each iteration the Hessian matrix H is constructed as previously described, and the direction vector s is determined from (21). The current method is unique in that a different method for the determination of the search direction vector is proposed.

Define the jth element of the vector of constraints $\hat{c}(x)$, as

$$\hat{c}_j(x) = c_i(x) \quad \text{if} \quad \begin{cases} i \in B \\ \text{or} \\ i \in B' \quad \text{and } c_i(x) < 0, \end{cases}$$

where $j = 1 \ldots \hat{m}$, $\hat{m} \leq m$. (23)

The $\hat{m}$ constraints in the vector $\hat{c}$ shall be referred to as active constraints. Using a similar notation for the corresponding gradients, define the Jacobian matrix of active constraints $\hat{G}$, and (14) becomes

$$\nabla J = \nabla L + 2r\hat{G}\,\hat{c} \tag{24}$$

where the definition (16) has been used. In like fashion, if we define

$$A = T + rU, \tag{25}$$

the expression for the Hessian matrix (17) is

$$H = A + 2r\,\hat{G}\hat{G}^T. \tag{26}$$

Combining (21), (24), and (26) one obtains the system of equations

$$(A + 2r\,\hat{G}\,\hat{G}^T)\,s \;=\; \nabla L + 2r\,\hat{G}\hat{c}. \tag{27}$$

Define the augmented system of equations

$$\begin{pmatrix} A & \hat{G} \\ \hat{G}^T & -\frac{1}{2r}I \end{pmatrix} \begin{pmatrix} s \\ \hat{\lambda} \end{pmatrix} = \begin{pmatrix} \nabla L + 2r\hat{G}\hat{c} \\ 0 \end{pmatrix} \tag{28}$$

where the vector $\hat{\lambda}$ has a dimension equal to the number of columns in $\hat{G}$ or the number of active constraints. I is the $\hat{m} \times \hat{m}$ identify matrix.

Let us show: (a) that the vector s obtained by solving (27) is equal to that obtained by solving (28), and (b) that the vector s obtained by solving (28) approaches the projected gradient direction as r becomes large. To indicate that the solution of (27) is equal to that obtained from (28), we employ the formula for the inverse of a matrix in terms of submatrices:

$$\begin{pmatrix} A & \hat{G} \\ \hat{G}^T & -\frac{1}{2r}I \end{pmatrix}^{-1} = \begin{pmatrix} C_{11} & C_{12} \\ C_{12}^T & C_{22} \end{pmatrix} \tag{29}$$

where

$$C_{11} = (A + 2r\,\hat{G}\,\hat{G}^T)^{-1}$$

$$C_{12} = 2r\,C_{11}\hat{G}$$

$$C_{22} = 4r^2\,\hat{G}^T\,C_{11}\,\hat{G} - 2rI.$$

Clearly from (28) and (29)

$$s = (A + 2r\,\hat{G}\hat{G}^T)^{-1} \quad (\nabla L + 2r\,\hat{G}\hat{c}) \tag{30}$$

which is the solution obtained from (27).

To investigate the limiting behavior it is convenient to expand (28) to form

$$As + \hat{G}\hat{\lambda} = \nabla L + 2r\,\hat{G}\hat{c} \tag{31}$$

and

$$\hat{G}^T s - \frac{1}{2r}\hat{\lambda} = 0. \tag{32}$$

It has been demonstrated by Fiacco and McCormick (Ref. 12), that

$$\lim_{r \to \infty} 2\,r\hat{c} = \hat{\lambda} \tag{33}$$

where $\hat{\lambda}$ is the vector of Lagrange multipliers. Assume that $U = 0$ (which is reasonable for large r), so that A is independent of r. Applying this limiting expression, (31) becomes

$$As = \nabla L. \tag{34}$$

Furthermore we can write (32) as

$$\hat{G}^T s = \hat{c}. \tag{35}$$

Making use of the definition of ∇L from (15) it follows that (34) and (35) can be written as

$$\begin{pmatrix} A & \hat{G} \\ \hat{G}^T & 0 \end{pmatrix} \begin{pmatrix} s \\ -\hat{\lambda} \end{pmatrix} = \begin{pmatrix} g \\ \hat{c} \end{pmatrix}. \tag{36}$$

Solution of this system using the same partitioning formulas used in (29) results in the standard projected gradient search direction.

Having considered the limiting behavior, it appears that there is some advantage to determining the search direction using the augmented system (28) instead of the system (27). Specifically, the condition number of the augmented system should approach a constant value as r becomes large, since the system (28) approaches the system (36). In contrast, Fletcher and McCann (Ref. 6) report that the condition number of the system (27) becomes infinite as r increases. Thus, at the expense of solving the larger system (28), the ill-conditioning associated with the solution of (27) can be avoided.

A second advantage of the proposed method is the fact that it is unnecessary to assume that A is positive definite. Consequently, the suggested approach is applicable when methods requiring inversion or Cholesky decomposition of A are not. For example, a linear objective function poses no difficulty in the new method, since it is not necessary that A have full rank.

Having discussed the general procedure for computing the search direction vector, let us make some observations pertinent to specific applications. First, it should be clear that the general algorithm is applicable to unconstrained optimization problems if we set $m = 0$ and $r = 0$. Nonlinear least squares problems can be solved using the algorithm if we set $f(x) = 0$, $\nabla L = 0$, and $r = 1$ for $m \geq n$. Furthermore, the algorithm can be used to satisfy the constraints in constrained optimization by posing a least squares problem with $\hat{m} \leq n$

and $\nabla L = 0$. With two exceptions, the search direction is determined from (28). The first exception involves the nonlinear least squares case when the matrix $U = 0$. In this instance, it is numerically preferable to apply the linear least squares algorithm directly to (35). The second exception occurs when solving constraints in a constrained optimization problem with $U = 0$ and $\hat{m} < n$. In this situation we solve (36), which is the limiting form of (28).

It should be noted at this point that any technique for solving linear systems can be applied to (28). For the class of problems of interest to the author, the computational cost of evaluating the function and gradient is far greater than the cost of solving the system of equations. Consequently, in the computer implementation, the system (28) is solved using the linear least squares procedure described in Ref. 13.

To summarize, an algorithm for finding the unconstrained minimum of the augmented objective function (11) has been outlined. The algorithm consists of a sequence of steps given by (22) in the directions defined by the vector s. The vector s is computed by solving the augmented system (28) subject to the exceptions noted above. The Hessian matrices are generated recursively using a rank-one formula as described in the references.

SECTION 3

APPLICATION TO CONSTRAINED OPTIMIZATION

Having developed a general unconstrained optimization algorithm, let us consider its use in an overall nonlinear programming method. It has been established that a point which minimizes the augmented objective function (11) also satisfies the Lagrangian condition (6) for specified penalty weight r and multipliers λ. It was indicated above that a point satisfying the constraint condition (8) could be determined by minimizing J, provided that we set $L(x, \lambda) = 0$ and $r = 1$. Since for $\hat{m} < n$ there are fewer constraints than variables, in general, there is no unique solution to the constraints. In order to make the point on the constraint surface unique, it is also required that the local quadratic approximation to the objective function be minimized. The search direction determined from (28) or (36) does, in fact, determine a unique point which is the exact solution when the constraints are linear and the objective function is quadratic. In fact, the accelerated multiplier method of Ref. 1 uses a single quadratic-linear step of the form (36). A principle difference between the new algorithm and that of Ref. 1 is the repeated use of the quadratic-linear steps until a point satisfying the constraints is located.

The unconstrained minimization algorithm requires a specified basis estimate B and specified multiplier estimates λ. Estimates of the multipliers are obtained by minimizing the error in the Kuhn-Tucker conditions. The multiplier estimates are then used to construct an estimate of the basis. The basis determination process computes multipliers and constructs a basis estimate at any point x. Details of the process are described in Ref. 1.

SECTION 4

THE GRADIENT PROJECTION MULTIPLIER ALGORITHM

The basic steps of the gradient projection multiplier algorithm are:

Step 1. Lagrangian Phase: For a fixed basis B^k, fixed multipliers λ^k, and fixed penalty weight r^k minimize the augmented objective function (11) using the unconstrained optimization algorithm given in Section 2. Call the solution $\bar{x}$.

Step 2. Basis Determination: Keeping $\bar{x}$ fixed, compute a new basis $\bar{B}$ and multipliers $\bar{\lambda}$ using the procedure described in Ref. 1.

Step 3. Constraint Phase: Beginning at $\bar{x}$ with the fixed basis $\bar{B}$, minimize the augmented objective function (11) with $L = 0$ and $r = 1$. Call the solution x^{k+1}. If $P(x^{k+1}) \neq 0$, constraints may be inconsistent.

Step 4. Basis Determination: Keeping x^{k+1} fixed, determine a new basis B^{k+1} and multipliers λ^{k+1} using the procedure described in Ref. 1. When checking for inconsistent constraints, B^{k+1} must be different than $\bar{B}$; if not, terminate.

Step 5. Convergence Test: $e(x, \lambda) < \delta_1$ and $\sigma_r < \delta_2$ where $e(x, \lambda)$ is the absolute error in the Kuhn-Tucker conditions (6) and (8), and σ_r is an estimate of the resolution error in the variables x and λ. (cf. Ref. 1).

Step 6. Penalty Weight Definition: If $B^{k+1} = \bar{B} = B^k$, keep penalty weight unchanged, i.e., set $r^{k+1} = r^k$. Otherwise, increase penalty weight (cf. Ref. 1).

Step 7. Update Information: Set $k = k+1$, $x^k = x^{k+1}$, $\lambda^k = \lambda^{k+1}$, $B^k = B^{k+1}$, etc. Return to Step 1.

A number of points regarding the implementation of the overall algorithm now deserve clarification. First, observe that current gradient and Hessian matrix information is used to initiate the different operations. For example, the Hessian matrices generated by the unconstrained algorithm during the Lagrangian phase can be used to initiate the penalty minimization in the constraint phase. Secondly, when there is an indication of inconsistent constraints, the basis determination procedure constructs a new basis by deleting all satisfied constraints from the old basis. If no constraints can be deleted, the algorithm terminates in an error mode.

The philosophy of the algorithm is to alternately satisfy the Lagrangian condition $\nabla L(x, \lambda) = 0$ and the constraint condition. During the constraint phase of the algorithm the search directions are computed using the assumption that the objective function is quadratic and the constraints are linear, resulting in the projected gradient directions. The penalty weight is not increased once the correct basis is determined; however, it is allowed to increase during the basis determination process. This procedure prevents cycling between two or more incorrect estimates of the basic set of constraints. Furthermore, the correct basis B^* is usually identified quickly because of the forced increase in the penalty weight.

SECTION 5

NUMERICAL EXPERIENCE

The algorithm described in the previous section has been implemented in a digital computer program. A rather extensive set of test problems have been solved using a CDC 7600 digital computer. This section presents the results of this numerical experience. Most of the problems have been drawn from the literature, and it is felt that the mathematical complexity and nonlinearity is fairly representative of the kind of problems encountered in practice. However, the problems do have two significant attributes which are distinct from those commonly encountered in engineering problems.

First, the function evaluation process is relatively inexpensive, and errors in the evaluation process are on the order of the machine accuracy. Similarly, accurate gradients can be obtained cheaply for the test problems. For practical problems, however, function evaluations can be quite costly and may contain inaccuracies significantly larger than the machine accuracy. (A typical trajectory optimization example is described in Ref. 14.) Because the function evaluations for these practical problems are so costly with respect to the computational expense of the optimization algorithm, we have used the number of function evaluations as a measure of algorithm effectiveness.

Secondly, the required gradient information is obtained by evaluating corresponding analytically derived partial derivative expressions. Practical problems, on the other hand, may require that gradient information be evaluated using numerical methods. For this reason, no attempt has been made to present an "equivalent" number of function evaluations, because such a quantity is highly dependent upon the numerical differentiation procedure. Such quantities as perturbation sizes and error tolerances can greatly influence the accuracy of numerical derivatives and consequently obscure the overall behavior of the optimization process. If gradient information must be obtained numerically, one can expect a two-fold degradation: (1) because each gradient evaluation will require one or more additional function evaluations, and (2) because inaccurate gradient information may necessitate more optimization iterations.

The test problems given in the appendix have been organized into four separate categories: (1) unconstrained, (2) nonlinear least squares, (3) equality constrained, and (4) inequality constrained. The results are presented in Tables 1 through 4 in condensed form for each of the categories. Specifically, we present the number of function and gradient evaluations required for convergence. For example, the first problem was solved by evaluating the objective function, constraints, and the corresponding gradient vectors at four points. For all problems, convergence is defined as in Ref. 1 with $\delta_1 = \delta_2 = 10^{-5}$, which guarantees five significant figures accuracy in the solution, and absolute satisfaction of the constraint and Lagrangian conditions to within 10^{-5}. The number of cycles of the algorithm is presented for the constrained examples. It is the author's opinion that despite the theoretical elegance of a general mathematical programming algorithm, one can not espouse its numerical effectiveness without solving a set of test problems at least as broad as those given.

SECTION 6
SUMMARY AND CONCLUSIONS

This paper describes a gradient projection-multiplier method for solving the general nonlinear programming problem. The algorithm poses a sequence of unconstrained optimization problems which are solved using a new projection-like formula to define the search directions. Ill conditioning of the search direction computation is avoided by using an expression which approaches the projected gradient direction for large penalty weights. The unconstrained algorithm is used to locate points where the Lagrangian condition $\nabla L(x, \lambda) = 0$ is satisfied by minimizing the augmented objective function. Points satisfying the constraints are located by applying the unconstrained algorithm to the penalty function. New estimates of the Lagrange multipliers and basis constraints are made at points satisfying the Lagrangian condition and the constraint condition. The penalty weight is increased only when necessary to prevent cycling. Although we do not prove quadratic convergence, numerical experience tends to confirm this assertion.

Table 1. Unconstrained Problems

Problem No.	Function and Gradient Evaluations
1	4
2	49
3	57
4	34
5	12
6	8
7	28
8	17
9	113
10	6
11	54

Table 2. Nonlinear Least Squares Problems

Problem No.	Function and Gradient Evaluations
1	6
2	27
3	11
4	9
5	2
6	15
7	6
8	22
9	12
10	10
11	3
12	7
13	15

Table 3. Equality Constrained Problems

Problem No.	Function and Gradient Evaluations	No. of Cycles
1	8	1
2	10	1
3	21	2
4	17	2
5	17	1
6	19	2
7	18	2
8	16	2
9	6	1
10	6 a	1
11	18b	1
12	27	1
13	188	2
14	46	2
15	60	2
16	141	2
17	39	1

a Algorithm terminated at a saddlepoint ($\nabla P(x^*) = 0$, although $P(x^*) \neq 0$, and $P(x) = 0$ does exist).

b Inconsistent constraints ($P(x^*) = 0$ does not exist).

Table 4. Inequality Constrained Problems

Problem No.	Function and Gradient Evaluations	No. of Cycles
1	21	1
2	5	1
3	22	1
4	28	2
5	18	1
6	58	2
7	50	4
8	50	2
9	32	3
10	28	2
11	43	1
12	43	3
13	77	3
14	27a	1
15	25	2
16	17a	1
17	40	3
18	48	3
19	21	2
20	65	4
21	13	1
22	14	2
23	33	2
24	48	3
25	38	2
26	29	2
27	81	1
28	318	1
29	9	1
30	79	1
31	86	1
32	29	1
33	43a	1
34	9	1

a Constraint gradients are linearly dependent at x^*.

REFERENCES

1. J. T. Betts, "An Accelerated Multiplier Method for Nonlinear Programming," Journal of Optimization Theory and Applications, Vol. 21, No. 2, Feb. 1977.

2. M.J.D. Powell, "A Method for Nonlinear Constraints in Minimization Problems," Optimization, edited by R. Fletcher, Academic Press, London and New York, 1969, p. 283.

3. M. R. Hestenes, "Multiplier and Gradient Methods," Journal of Optimization Theory and Applications, Vol. 4, No. 5, 1969.

4. R. Fletcher, "Minimizing General Functions Subject to Linear Constraints," Numerical Methods for Nonlinear Optimization, edited by F. A. Lootsma, Academic Press, London and New York, 1972.

5. B.A. Murtagh and R.W.H. Sargent, "A Constrained Minimization Method with Quadratic Convergence," Optimization, edited by R. Fletcher, Academic Press, London and New York, 1969, p. 215.

6. R. Fletcher and A.P. McCann, "Acceleration Techniques for Nonlinear Programming," Optimization, edited by R. Fletcher, Academic Press, London and New York, 1969, p. 203.

7. M.C. Biggs, "Constrained Minimization Using Recursive Equality Quadratic Programming," Numerical Methods for Nonlinear Optimization, edited by F.A. Lootsma, Academic Press, London and New York, 1972, p. 411.

8. W. Murray, "An Algorithm for Constrained Minimization," Optimization, edited by R. Fletcher, Academic Press, London and New York, 1969, p. 247.

9. J. T. Betts, "An Improved Penalty Function Method for Solving Constrained Parameter Optimization Problems," Journal of Optimization Theory and Applications, Vol. 16, No. 1-2, 1975.

10. J. T. Betts, "Solving the Nonlinear Least Square Problem: Application of a General Method," Journal of Optimization Theory and Applications, Vol. 18, No. 4, 1976.

11. J. T. Betts, An Accelerated Multiplier Method for Nonlinear Programming, The Aerospace Corporation, El Segundo, California, Report No. TR-0075(5901-03)-5, 1974.

12. A. V. Fiacco and G. P. McCormick, Nonlinear Programming: Sequential Unconstrained Minimization Techniques, John Wiley, New York, 1968.

13. R. J. Hanson and C. L. Lawson, "Extensions and Applications of the Householder Algorithm for Solving Linear Least Squares Problems," Mathematics of Computation, Vol. 23, No. 108, 1969.

14. J. T. Betts and A. D. Hemenover, "Optimal Three Burn Orbit Transfer," to be published AIAA Journal.

15. R. Fletcher and M.J.D. Powell, "A Rapidly Convergent Descent Method for Minimization," Computer Journal, Vol. 6, No. 2, 1963.

16. H. H. Rosenbrock, "An Automatic Method for Finding the Greatest or Least Value of a Function" Computer Journal, Vol. 3, No. 175, 1960.

17. A. A. Leon, "A Comparison Among Eight Known Optimizing Procedures," Recent Advances in Optimization Techniques, edited by A. Lavi and T. P. Vogl, John Wiley, New York, pp. 23-42.

18. E. M. L. Beale, On An Iterative Method for Finding a Local Minimum of a Function of More Than One Variable, Statistical Techniques Research Group, Princeton University, Tech. Report 25.

19. R. H. Wampler, "An Evaluation of Linear Least Squares Computer Programs," Journal of Research of National Bureau of Standards, B. Math. Sci., Vol. 73B, No. 2, 1969.

20. A. R. Colville, A Comparative Study on Nonlinear Programming Codes, IBM New York Scientific Center, New York, Report No. 320-2949, 1968.

21. D. M. Himmelblau, Applied Nonlinear Programming, McGraw-Hill, New York, 1972.

APPENDIX

NONLINEAR PROGRAMMING TEST PROBLEMS

This appendix presents the set of test problems used to assess the effectiveness of the nonlinear programming algorithm described. The problems are organized into four categories: (1) unconstrained problems, (2) nonlinear least squares problems, (3) equality constrained problems, and (4) inequality constrained problems.

When no information is given to the contrary, one can assume all quantities used by the numerical processes are scaled (in the sense described in the Appendix of Ref. 9) in the range $-20 \le x \le 20$. Unless noted otherwise, the initial penalty weight is $r^o = 1$, and the initial basis is assumed to be empty, i.e., $B^o = \{\Phi\}$. The points x^* presented are converged values where convergence is defined in Section 7 of Ref. 1, with $\delta_1 = \delta_2 = 10^{-5}$. When the exact solution is known, its value is presented following the computationally obtained value. To conserve space, problem statements appearing elsewhere in consistent notation are merely referenced.

A. 1 UNCONSTRAINED PROBLEMS

1. Ref. 15

Minimize $f(x) = x_1^2 - 2x_1x_2 + 2x_2^2$

$-50 \leq x \leq 50$

$x^o = (4, 2)$

$x^* = (2.1316 \times 10^{-14}, 1.0658 \times 10^{-14})$; $(0, 0)$

$f^* = 2.2719 \times 10^{-28}$; 0

no scaling.

2. Ref. 16

Minimize $f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$

$x^o = (-1.2, 1)$

$x^* = (1., 1.)$; $(1, 1)$

$f^* = 6.6421 \times 10^{-18}$; 0

no scaling.

3. Ref. 15

Minimize $f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 + 10(x_1 - x_4)^4$

$-50 \leq x \leq 50$

$x^o = (3, -1, 0, 1)$

$x^* = (1.8159 \times 10^{-4}, -1.8159 \times 10^{-5}, 9.398 \times 10^{-5}, 9.3980 \times 10^{-5})$; $(0, 0, 0, 0)$

$f^* = 2.3943 \times 10^{-15}$; 0

no scaling.

4. Ref. 15

$$\text{Minimize} \quad f(x) = 100\left[(x_3 - 10\theta)^2 + (r-1)^2\right] + x_3^2$$

where

$$2\pi\theta = \begin{cases} \tan^{-1}(x_2/x_1) & x_1 > 0 \\ \pi + \tan^{-1}(x_2/x_1) & x_1 < 0 \end{cases}$$

$$r = (x_1^2 + x_2^2)^{\frac{1}{2}}$$

$-5 \le x_1, x_2 \le 5$

$-2.5 \le x_3 \le 7.5$

$x^o = (-1, 0, 0)$

$x^* = (1., 5.2171 \times 10^{-10}, 6.0920 \times 10^{-10}) \quad ; \quad (1, 0, 0)$

$f^* = 7.9033 \times 10^{-18} \quad ; \quad 0$

no scaling.

5. Ref. 17

$$\text{Minimize} \quad f(x) = (x_2 - x_1^2)^2 + (1 - x_1)^2$$

$-100 \le x \le 100$

$x^o = (-2, -2)$

$x^* = (9.9999 \times 10^{-1}, 9.9999 \times 10^{-1}) \quad ; \quad (1, 1)$

$f^* = 2.4809 \times 10^{-16} \quad ; \quad 0$

no scaling.

6. Ref. 17

$$\text{Minimize} \quad f(x) = (x_2 - x_1^2)^2 + 100(1 - x_1)^2$$

$-100 \le x \le 100$

$x^o = (1, 5)$

$x^* = (1., 1.) \quad ; \quad (1, 1)$

$f^* = 1.0172 \times 10^{-21} \quad ; \quad 0$

no scaling.

7. Ref. 17

$$\text{Minimize} \quad f(x) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2$$

$-100 \le x \le 100$

$x^o = (-1.2, 1)$

$x^* = (9.9999 \times 10^{-1}, 9.9999 \times 10^{-1}) \quad ; \quad (1, 1)$

$f^* = 2.0912 \times 10^{-13} \quad ; \quad 0$

no scaling.

8. Ref. 18

$$\text{Minimize} \quad f(x) = \sum_{i=1}^{3} \left[a_i - x_1 (1 - x_2^i) \right]^2$$

where

$a_1 = 1.5 \quad , \quad a_2 = 2.25 \quad , \quad a_3 = 2.625$

$-100 \le x \le 100$

$x^o = (8, .2)$

$x^* = (3, .5) \quad ; \quad (3, .5)$

$f^* = 1.7257 \times 10^{-20} \quad ; \quad 0$

no scaling.

9. Ref. 20

$$\text{Minimize } f(x) = 100\,(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90\,(x_4 - x_3^2)^2$$

$$+ (1 - x_3)^2 + 10.1\,[(x_2 - 1)^2 + (x_4 - 1)^2]$$

$$+ 19.8\,(x_2 - 1)(x_4 - 1)$$

$-10 \leq x \leq 10$

$x^o = (-3, -1, -3, -1)$

$x^* = (1, 1, 1, 1)$

$f^* = .1969 \times 10^{-16}; 0$

no scaling.

10. Ref. 21

$$\text{Minimize } f(x) = \sum_{i=1}^{10} \left\{ [\ln (x_i - 2)]^2 + [\ln (10 - x_i)]^2 \right\}$$

$$- \left(\prod_{i=1}^{10} x_i\right)^{0.2}$$

$2.001 \leq x \leq 9.999$

$x_1^o = 9 \quad i = 1, \ldots, 10$

$x_i^* = 9.3503 \quad i = 1, \ldots 10.$

$f^* = -45.778$

no scaling.

11. Ref. 21

$$\text{Minimize } f(x) = \sum_{i=1}^{99} \left[\exp - \frac{(u_i - x_2)^{x_3}}{x_1} - .01i \right]^2$$

where $u_i = 25 + (-50 \ln .01i)^{2/3}$

$.1 \le x_1 \le 101$

$0 \le x_2 \le 25.6$

$0 \le x_3 \le 5.$

$x^o = (100,\ 12.5,\ 3)$

$x^* = (50,\ 25,\ 1.5)$

$f^* = .52009 \times 10^{-17};\ 0$

A.2 NONLINEAR LEAST SQUARES PROBLEMS

Problems. 1-10: Problems 1-10 in Ref 10.

Problem 11. Ref. 19

$$\text{Minimize } f(x) = \sum_{i=1}^{21} c_i^2(x)$$

where $t_i = i - 1$

$$y_i = 1 + t_i + t_i^2 + t_i^3 + t_i^4 + t_i^5$$

$$c_i(x) = y_i - (x_1 + x_2 t_i + x_3 t_i^2 + x_4 t_i^3 + x_5 t_i^4 + x_6 t_i^5)$$

$$x_i^o = 0 \qquad i = 1, \ldots 6$$

$$x^* = (1.00000,\ 1.00000,\ 1.00000,\ 1.00000,\ 1.00000,\ 1.00000)$$

$$f^* = .13777 \times 10^{-14}$$

No scaling

Problem 12.

$$\text{Minimize } f(x) = \sum_{i=1}^{13} c_i^2(x)$$

where $c_i(x) = \sqrt{w_i}\ \{ y_i - [x_1 u_i + \exp(x_2 v_i)] \}$

and u_i, v_i, w_i, and y_i are given in Table A-1.

TABLE A-1

i	u_i	v_i	w_i	y_i
1	0	0	100	2.93
2	0	1	100	1.95
3	0	2	100	.81
4	0	3	100	.58
5	1	0	50	5.90
6	1	1	50	4.74
7	1	2	50	4.18
8	1	2	50	4.05
9	2	0	25	9.03
10	2	1	25	7.85
11	2	2	25	7.22
12	2.5	2	10	8.50
13	2.9	1.8	10	9.81

$-10 \le x \le 10$

$x^o = (3.01, -.51)$

$x^* = (3.5593, -.13414)$

$f^* = 651.147$

No Scaling.

Problem 13. Problem No. 11 in Ref. 10.

A. 3 EQUALITY CONSTRAINED PROBLEMS

Problems 1-12: Problems 1-12 in Ref. 1.

13.

Minimize $f(x) = .001\ x_1 + x_2$

$c_1(x) = 10^5\ (x_2 - x_1^{\ 2}) = 0$

$-100 \leq x \leq 100$

$x^o = (1,\ 1)$

$x^* = (-.49999 \times 10^{-3},\ .25000 \times 10^{-6})$

$f^* = -.249999 \times 10^{-6}$

No scaling

14.

Minimize $f(x) = .001\ x_1 + x_2$

$c_1(x) = -1000\ x_1^{\ 2} - 100\ x_2^{\ 2} + x_3 = 0$

$c_2(x) = 100\ x_1^{\ 2} + 400\ x_2^{\ 2} + x_3 - .01 = 0$

$-10 \leq x \leq 10$

$x^o = (1,\ 1,\ 1)$

$x^* = (.20908 \times 10^{-5},\ .44721 \times 10^{-2},\ .20000 \times 10^{-2})$

$f^* = .44721 \times 10^{-2}$

No scaling.

15. Ref. 21

$$\text{Minimize } f(x) = \sum_{i=1}^{10} x_i \left(a_i + \ln \frac{x_i}{\sum_{j=1}^{10} x_j} \right)$$

where $a_1 = -6.089$ $a_2 = 17.164$ $a_3 = -34.054$

$a_4 = -5.914$ $a_5 = -24.721$ $a_6 = -14.986$

$a_7 = -24.100$ $a_8 = -10.708$ $a_9 = -26.662$

$a_{10} = -22.179$

$c_1(x) = x_1 + 2x_2 + 2x_3 + x_6 + x_{10} - 2 = 0$

$c_2(x) = x_4 + 2x_5 + x_6 + x_7 - 1 = 0$

$c_3(x) = x_3 + x_7 + x_8 + 2x_9 + x_{10} - 1 = 0$

$1. \times 10^{-5} \leq x \leq 10$

$x_i^o = .1 \quad i = 1, \ldots, 10$

$x^* = (.40668 \times 10^{-1}, .14773, .78315, .14142 \times 10^{-2}, .48524,$
$.69317 \times 10^{-3}, .27399 \times 10^{-1}, .17947 \times 10^{-1}, .37314 \times 10^{-1},$
$.96871 \times 10^{-1})$

$f^* = -47.761$

no scaling.

16. Ref. 21

$$\text{Minimize } f(x) = \sum_{i=1}^{10} \left\{ e^{x_i} \left[a_i + x_i - \ln\left(\sum_{j=1}^{10} e^{x_j}\right) \right] \right\}$$

where the a_i are given in Problem 15.

$$c_1(x) = e^{x_1} + 2e^{x_2} + 2e^{x_3} + e^{x_6} + e^{x_{10}} - 2 = 0$$

$$c_2(x) = e^{x_4} + 2e^{x_5} + e^{x_6} + e^{x_7} - 1 = 0$$

$$c_3(x) = e^{x_3} + e^{x_7} + e^{x_8} + 2e^{x_9} + e^{x_{10}} - 1 = 0$$

$$-100 \leq x \leq 100$$

$$x_i^o = -2.3 \qquad i = 1, \ldots, 10$$

$$x^* = (-3.2024, -1.9123, -.24441, -15.670, -.72166,$$
$$-7.2736, -3.5965, -4.0206, -3.2885, -2.3344)$$

$$f^* = -47.760$$

Constraint scaling: $w_i = 10$, $i = 1, 2, 3$.

17. Ref. 21

$$\text{Minimize } f(x) = 1000 - x_1^2 - 2x_2^2 - x_3^2 - x_1x_2 - x_1x_3$$

$$c_1(x) = x_1^2 + x_2^2 + x_3^2 - 25 = 0$$

$$c_2(x) = 8x_1 + 14x_2 + 7x_3 - 56 = 0$$

$$0 \leq x \leq 100$$

$$x^o = (2, 2, 2)$$

$$x^* = (3.5121, .21698, 3.5522)$$

$$f^* = 961.715$$

No scaling.

A.4 INEQUALITY CONSTRAINED PROBLEMS

Problems 1-26: Problems 1-26 in Ref 1.

27. Problem 27 in Ref. 1 except with

$B^o = \{\Phi\}$

Constraint scaling:

$w_0 = 10^{-6}$

$w_i = 1 \quad i = 1, 2, 3, 4, 5, 7$

$w_6 = 10$

$w_8 = w_9 = 100$

$w_i = 10^{-5} \quad i = 10, 11, 12, 13, 14, 15$

28. Ref. 20

$$\text{Maximize } f(x) = \sum_{i=6}^{15} \beta_{i-5} x_i - \sum_{j=1}^{5} \sum_{i=1}^{5} \gamma_{ij} x_i x_j - 2\sum_{i=1}^{5} \delta_i x_i^3$$

$$c_i(x) = x_i \geq 0 \qquad i = 1, \ldots 15$$

$$c_j(x) = \epsilon_{j-15} + 3\,\delta_{j-15}\, x_{j-15}^2 + 2\sum_{i=1}^{5} \gamma_{i,\, j-15}\, x_i$$

$$-\sum_{i=6}^{15} \alpha_{i-5,\, j-15}\, x_i \geq 0 \qquad j = 16, \ldots 20.$$

where the coefficients α, β, γ, δ, and ϵ are given in Problem 17 of Ref 1.

$$-10 \leq x_i \leq 20 \qquad i = 1, \ldots, 11$$

$$-100 \leq x_{12} \leq 100$$

$-10 \le x_i \le 10$ $\quad$ $i = 13, 14, 15.$

$x_i^o = .0001$ $\quad$ $i = 1, \ldots, 15; i \neq 12$

$x_{12}^o = 60$

$x^* = (.30000, .33346, .40000, .42832, .22397,$

$0., -.56843 \times 10^{-13}, 5.1740, 0., 3.06111,$

$11.8396, 0., 0., .10391, 0.)$

$f^* = -32.3486$

Constraint scaling:

$w_i = 10$ for $i = 6, 12, 13$

$w_i = 1$ otherwise

29. Ref. 21

Minimize $f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$

$c_1(x) = x_1 - 2x_2 + 1 = 0$

$c_2(x) = -\frac{x_1^2}{4} - x_2^2 + 1 \ge 0$

$-10 \le x \le 10$

$x^o = (2, 2)$

$x^* = (.82287, .91143)$

$f^* = 1.3934$

no scaling.

30. Ref. 21

Maximize $f(x) = 75.196 - 3.8112\, x_1 + .12694\, x_1^2$

$-2.0567 \times 10^{-3}\, x_1^3 + 1.0345 \times 10^{-5}\, x_1^4 - 6.8306\, x_2$

$+.030234\, x_1 x_2 - 1.28134 \times 10^{-3}\, x_2 x_1^2$

$+ 3.5256 \times 10^{-5}\, x_2 x_1^3 - 2.266 \times 10^{-7}\, x_2 x_1^4$

$+.25645\, x_2^2 - 3.4604 \times 10^{-3}\, x_2^3 + 1.3514 \times 10^{-5}\, x_2^4$

$-(28.106)\,(x_2 + 1)^{-1} - 5.2375 \times 10^{-6}\, x_1^2\, x_2^2$

$-6.3 \times 10^{-8}\, x_1^3\, x_2^2 + 7 \times 10^{-10}\, x_1^3\, x_2^3$

$+ 3.4054 \times 10^{-4}\, x_1 x_2^2 - 1.6638 \times 10^{-6}\, x_1 x_2^3$

$-2.8673 \exp(.0005\, x_1 x_2)$

$$c_1(x) = x_1 x_2 - 700 \geq 0$$

$$c_2(x) = x_2 - 5 \left(\frac{x_1}{25}\right)^2 \geq 0$$

$$c_3(x) = (x_2 - 50)^2 - 5(x_1 - 55) \geq 0$$

$$c_4(x) = 75 - x_1 \geq 0$$

$$c_5(x) = 65 - x_2 \geq 0$$

$$0 \leq x \leq 200$$

$$x^o = (90,\ 10)$$

$$x^* = (75,\ 65)$$

$$f^* = 58.903$$

no scaling.

31. Ref. 20. p. 31. (Also Ref. 21, p. 401)

32. Ref. 21

$$\text{Minimize } f(x) = \sum_{i=1}^{19} (y_i - \hat{y}_i)^2$$

where
$$y_i = x_3\, \beta^{x_2} \left(\frac{x_2}{6.2832}\right)^{1/2} \left(\frac{a_i}{7.658}\right)^{(x_2-1)}$$

$$\cdot \exp\left(x_2 - \beta\, \frac{a_i\, x_2}{7.658}\right) \left(1 + \frac{1}{12 x_2}\right)^{-1}$$

$$+ (1 - x_3) \left(\frac{\beta}{x_4}\right)^{x_1} \left(\frac{x_1}{6.2832}\right)^{1/2} \left(\frac{a_i}{7.658}\right)^{(x_1-1)}$$

$$\cdot \exp\left(x_1 - \beta\, \frac{a_i\, x_1}{7.658 x_4}\right) \left(1 + \frac{1}{12 x_1}\right)$$

$$\beta = x_3 + (1-x_3)x_4$$

and the a_i and $\hat{y}_i$ are given in Table A-2

$$c_1(x) = x_3 + (1-x_3)x_4 \geq 0$$

$$1 \times 10^{-10} \leq x \leq 20$$

$$x^o = (2,\ 4,\ .04,\ 2)$$

$$x^* = (12.277,\ 4.6318,\ .31286,\ 2.0293)$$

$$f^* = .74985 \times 10^{-2}$$

TABLE A-2

i	a_i	$\hat{y}_i$	i	a_i	$\hat{y}_i$
1	.1	.00189	11	10	.702
2	1	.1038	12	11	.528
3	2	.268	13	12	.385
4	3	.506	14	13	.257
5	4	.577	15	14	.159
6	5	.604	16	15	.0869
7	6	.725	17	16	.0453
8	7	.898	18	17	.01509
9	8	.947	19	18	.00189
10	9	.845			

33. Ref. 21

Maximize $f(x) = .5\,(x_1x_4 - x_2x_3 + x_3x_9 - x_5x_9 + x_5x_8 - x_6x_7)$

$$c_1(x) = 1 - x_3^2 - x_4^2 \geq 0$$

$$c_2(x) = 1 - x_9^2 \geq 0$$

$$c_3(x) = 1 - x_5^2 - x_6^2 \geq 0$$

$$c_4(x) = 1 - x_1^2 - (x_2 - x_9)^2 \geq 0$$

$$c_5(x) = 1 - (x_1 - x_5)^2 - (x_2 - x_6)^2 \geq 0$$

$$c_6(x) = 1 - (x_1 - x_7)^2 - (x_2 - x_8)^2 \geq 0$$

$$c_7(x) = 1 - (x_3 - x_5)^2 - (x_4 - x_6)^2 \geq 0$$

$c_8(x) = 1 - (x_3 - x_7)^2 - (x_4 - x_8)^2 \geq 0$

$c_9(x) = 1 - x_7^2 - (x_8 - x_9)^2 \geq 0$

$c_{10}(x) = x_1 x_4 - x_2 x_3 \geq 0$

$c_{11}(x) = x_3 x_9 \geq 0$

$c_{12}(x) = - x_5 x_9 \geq 0$

$c_{13}(x) = x_5 x_8 - x_6 x_7 \geq 0$

$c_{14}(x) = x_9 \geq 0$

$-2 \leq x \leq 2$

$x_i^o = 1 \quad i = 1, \ldots, 9$

$x^* = (.91878, .39476, .11752, .99307, .91878,$
$.39476, .11752, .99307, -.60445 \times 10^{-14})$

$f^* = .86602$

no scaling.

34. Ref. 21

Minimize $f(x) = (x_1 - 2)^2 + (x_2 - 1)^2$

$c_1(x) = -x_1^2 + x_2 \geq 0$

$c_2(x) = -x_1 - x_2 + 2 \geq 0$

$x^o = (2, 2)$

$x^* = (1, 1)$

$f^* = 1$

no scaling.